AMENDMENTS TO THE CLAIMS

The listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently Amended) A method for load balancing code execution, said method comprising:

   compiling source code, the compiling resulting in byte code;

   retrieving the byte code at runtime using a runtime loader; ~~a code segment from a plurality of code segments;~~

   in response to retrieving the byte code, using the runtime loader to identify ~~identifying~~ a processor type from a plurality of heterogeneous processor types to execute the byte code ~~code segment~~;

   in response to identifying the processor type, using the runtime loader to translate the byte code to object code; and

   loading the object code ~~code segment~~ into a processor that corresponds to the identified processor type.

2. (Canceled)

3. (Currently Amended) The method as described in claim [[2]] 1 wherein the byte code includes a byte code type, the byte code type selected from the group consisting of Java, XML, HTML, Shader, and Script.

4. (Currently Amended) The method as described in claim [[2]] 1 further comprising:

   ~~compiling source code, the compiling resulting in the byte code;~~

   determining whether to store a pointer in a byte code file, the pointer including a stored location that corresponds to the byte code;

storing the pointer in the byte code file in response to the determination;

storing the byte code at the stored location in response to the determination; and

performing the retrieving using the pointer, wherein the retrieving includes analyzing the stored location and retrieving the byte code in response to the analyzing.

5. (Currently Amended) The method as described in claim 1 wherein the identifying includes using the runtime loader to analyze ~~analyzing~~ the availability of each of the plurality of heterogeneous processor types, and wherein the analyzing includes retrieving a loading factor for each of the plurality of heterogeneous processor types, which corresponds to the availability of each of the plurality of heterogeneous processor types.

6. (Currently Amended) The method as described in claim 1 wherein the identifying further comprises:

detecting, using the runtime loader at runtime, one or more operations included in the byte code ~~code segment~~; and

matching, using the runtime loader at runtime, one or more of the operations with one of the processor types from the plurality of heterogeneous processor types.

7. (Currently Amended) The method as described in claim 1 wherein the identifying further comprises:

determining, using the runtime loader at runtime, whether the byte code ~~code segment~~ includes a program directive corresponding to one of the plurality of processors; and

matching, using the runtime loader at runtime, one or more of the operations with one of the processor types from the plurality of heterogeneous processor types in response to the determination.

8. (Currently Amended) An information handling system comprising:

a plurality of heterogeneous processors;

a memory accessible by the heterogeneous processors;

one or more nonvolatile storage devices accessible by the heterogeneous processors; and

a code execution load balancing tool for load balancing code execution, the code execution load balancing tool comprising software code effective to:

compile source code, the compiling resulting in byte code;

retrieve the byte code at runtime from the memory using a runtime loader; ~~a code segment from a plurality of code segments located in the memory;~~

in response to retrieving the byte code, using the runtime loader to identify a processor type from a plurality of heterogeneous processor types to execute the byte code ~~code segment~~;

in response to identifying the processor type, use the runtime loader to translate the byte code to object code; and

load the object code ~~code segment~~ into one of the processors from the plurality of heterogeneous processors that corresponds to the identified processor type.

9. (Canceled)

10. (Currently Amended) The information handling system as described in claim [[9]] 8 wherein the byte code includes a byte code type, the byte code type selected from the group consisting of Java, XML, HTML, Shader, and Script.

11. (Currently Amended) The information handling system as described in claim [[9]] 8 wherein the software code is further effective to:

~~compile source code, the compiling resulting in the byte code;~~

determine whether to store a pointer in a byte code file, the pointer including a stored location that corresponds to the byte code;

store the pointer in the byte code file in the memory in response to the determination;

store the byte code at the stored location in the memory in response to the determination; and

perform the retrieving using the pointer, wherein the retrieving includes analyzing the stored location and retrieving the byte code from the memory in response to the analyzing.

12. (Currently Amended) The information handling system as described in claim 8 wherein the identifying includes using the runtime loader to analyze ~~analyzing~~ the availability of each of the plurality of heterogeneous processor types, and wherein the analyzing includes retrieving a loading factor for each of the plurality of heterogeneous processor types, which corresponds to the availability of each of the plurality of heterogeneous processor types.

13. (Currently Amended) The information handling system as described in claim 8 wherein the software code is further effective to:

detect, using the runtime loader at runtime, one or more operations included in the byte code ~~code segment~~; and

match, using the runtime loader at runtime, one or more of the operations with one of the processor types from the plurality of heterogeneous processor types.

14. (Currently Amended) A computer program product stored on a computer operable media for load balancing code execution, said computer program product comprising:

means for compiling source code, the compiling resulting in byte code;

means for retrieving the byte code at runtime using a runtime loader; ~~a code segment from a plurality of code segments;~~

in response to retrieving the byte code, means for using the runtime loader to identify ~~identifying~~ a processor type from a plurality of heterogeneous processor types to execute the byte code ~~code segment~~;

in response to identifying the processor type, means for using the runtime loader to translate the byte code to object code; and

means for loading the object code ~~code segment~~ into a processor that corresponds to the identified processor type.

15. (Canceled)

16. (Currently Amended) The computer program product as described in claim [[15]] 14 wherein the byte code includes a byte code type, the byte code type selected from the group consisting of Java, XML, HTML, Shader, and Script.

17. (Currently Amended) The computer program product as described in claim [[15]] 14 further comprising:

~~means for compiling source code, the compiling resulting in the byte code;~~

means for determining whether to store a pointer in a byte code file, the pointer including a stored location that corresponds to the byte code;

means for storing the pointer in the byte code file in response to the determination;

means for storing the byte code at the stored location in response to the determination; and means for performing the retrieving using the pointer, wherein the retrieving includes analyzing the stored location and retrieving the byte code in response to the analyzing.

18. (Currently Amended) The computer program product as described in claim 14 wherein the identifying includes using the runtime loader to analyze ~~analyzing~~ the availability of each of the plurality of heterogeneous processor types, and wherein the analyzing includes retrieving a loading factor for each of the plurality of heterogeneous processor types, which corresponds to the availability of each of the plurality of heterogeneous processor types.

19. (Currently Amended) The computer program product as described in claim 14 wherein the identifying further comprises:

means for detecting, using the runtime loader at runtime, one or more operations included in the byte code ~~code segment~~; and

means for matching, using the runtime loader at runtime, one or more of the operations with one of the processor types from the plurality of heterogeneous processor types.

20. (Currently Amended) The computer program product as described in claim 14 wherein the identifying further comprises:

means for determining, using the runtime loader at runtime, whether the byte code ~~code segment~~ includes a program directive corresponding to one of the plurality of heterogeneous processor types; and

means for matching, using the runtime loader at runtime, one or more of the operations with one of the processor types from the plurality of heterogeneous processor types in response to the determination.